New Jersey Institute of Technology
Digital Commons @ NJIT

Theses | Theses and Dissertations

Fall 1993

Authentication for mobile computing

Andreas Keppler
New Jersey Institute of Technology

Follow this and additional works at: https://digitalcommons.njit.edu/theses

 Part of the Computer Sciences Commons

Recommended Citation
Keppler, Andreas, "Authentication for mobile computing" (1993). *Theses*. 1218.
https://digitalcommons.njit.edu/theses/1218

This Thesis is brought to you for free and open access by the Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

NJIT New Jersey's Science & Technology University

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

Authentication
for
Mobile Computing

by
Andreas Keppler

Host mobility is becoming an increasingly important feature with the recent arrival of laptop and palmtop computers, the development of wireless network interfaces and the implementation of global networks. Unfortunately, this mobile environment is also much more vulnerable to penetration by intruders. A possible means of protection can be authentication. This guarantees the identity of a communication peer.

This thesis studies the constraints imposed on the mobile environment with respect to authentication. It compares the two prevailing authentication mechanisms, Kerberos and SPX, and tries to make suggestions of how a mechanism can be adapted to the mobile environment.

AUTHENTICATION FOR MOBILE COMPUTING

by
Andreas Keppler

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

Department of Computer and Information Science

January 1994

APPROVAL PAGE

Authentication
for
Mobile Computing

Andreas Keppler

Dr. Bruce Parker, Thesis Advisor — Date
Assistant Professor of Computer and Information Science, NJIT

Dr. Wilhelm Rossak, Committee Member — Date
Assistant Professor of Computer and Information Science
Director of System Integration Laboratory, NJIT

Dr. Alexander David Stoyenko, Committee Member — Date
Associate Professor of Computer and Information Science
Director of Real-Time Computing Laboratory, NJIT

BIOGRAPHICAL SKETCH

Author: Andreas Keppler

Degree: Master of Science in Computer Science

Date: January 1994

Undergraduate and Graduate Education:

- Master of Science in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1994
- Bachelor of Science in Computer Science,
  Fachhochschule fuer Technik Mannheim, Germany, 1992

Major: Computer Science

This work is dedicated to
Pee and our future

ACKNOWLEDGMENT

The author wishes to thank his thesis advisor, Dr. Bruce Parker, for his guidance and support which went beyond the call of duty with this thesis. He also grateful for the patience and the time spent by the committee members, Dr. Alex Stoyenko and Dr. Wilhelm Rossak.

The author wishes to express his sincere and heartfelt gratitude to the "Fulbright Kommission", which, by its financial support, gave him the opportunity to participate in the "Master Program".

Special thanks to Andrei Bergners and Diana Bozian for their proofreading, and especially to Andrei for his support as a friend besides the academic work.

Also special recognition to my girlfriend Pee for her direct and indirect encouragement through all my endeavors. She is and will ever be a motivation for me.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The evolution of computers has followed several parallel tracks. After the advent of bulky, expensive, and rare "mainframe" computers, they quickly evolved to timesharing systems, thereby bringing computational facilities to those individuals with access to a terminal. The development of the microprocessor led, in turn, to the concept of the personal computer (PC). Further advances in technology have made it possible to pack a personal computer, ranging in computational power from a simple "PC" to a fairly powerful workstation, into a small, lightweight, portable device, which satisfies the user's desire to access computational resources at their fingertips, and enable them to carry those resources with them.

Besides the increase in functionality of computers, the evolving availability of wireless network options support this striving for mobility. The wide–spread grid of wireless cells provides the user with continuous network access regardless of their location. A user is no longer restricted to stay in reach of a tap for a network connection.

This recent emergence in technology which supports user mobility has also prompted new security requirements and concerns. This is mainly due to the lack of physical protection mechanisms as in the traditional fixed–topology, static–user environment. The main problems are to prevent illegal access (fraud), impersonating, and eavesdropping in the mobile environment. Wireless communications are even more vulnerable to these kinds of attacks. Since one can not exhaustively search the transmission medium for intruders, it is easier for them to penetrate the network. Moreover, with increasing size of the cells due to the improving strength of the sending and receiving devices the space for an intruder to hide increases, too.

Authentication might be a means of protection against the first two kinds of attacks. It is the process of identifying an individual, as well as the verification of this pretended identity. This assures a communication peer about the trustworthiness of his partner. The third type of attack, eavesdropping, can not be remedied with mere authentication. Therefore, it is necessary to encrypt messages sent over the network. However, authentication can at least provide auxiliary function. In addition to authenticating an individual, the authentication exchange can also serve the purpose of securely providing communication parties with a session key to protect subsequent communications.

There already exists implemented authentication mechanisms. The predominant mechanisms include Kerberos and SPX. Kerberos is based on a private key cryptosystem, whereas SPX is built upon a public key cryptosystem. The mobile environment has some specific aspects to consider. This is in regard to the limitations of the devices used, like computing power and memory, as well as the unreliability of the transmission medium.

The next chapter gives an introduction of what authentication is, and how it works. Following, is a reflect on mobility, considering the constraints in the mobile environment and the current work in this area. Chapters 4 and 5 present an overview of the two prevailing authentication mechanism, Kerberos and SPX, respectively. Next is a comparison of these two mechanisms, considering the advantages and disadvantages of each one in different areas. Chapter 7 tries to make assumptions of how a mechanism could be adapted to the mobile environment. The last chapter provides an outlook on further work in this area and a conclusion.

CHAPTER 2

AUTHENTICATION

2.1 Threats

Networks are exposed to various attacks from *enemies* who may have access to the network or to the attached machines. These intruders may try to steal, or change messages, in order to obtain information or impersonate an other user. The potential threats for the transfer of information are:

- **Identity interception**: the identity of one or more of the users involved in a communication is observed for misuse. Possible remedial measures to protect against this unauthorized disclosure is the use of encryption of the data to be transferred.

- **Masquerade**: the pretense by a user to be a different user in order to gain access to information or to acquire additional privileges. To prevent this kind of attack, it is possible to require the user to prove his knowledge of a secret. Either by presenting it, or by performing a certain operation, only the owner of the secret is able to. An example would be the knowledge of a password for most of the login procedures.

- **Replay**: the recording and subsequent replay of a communication at a later date. Replay detection can be provided by using timestamps, combined with remembering previously sent messages, or by challenge and response exchanges.

- **Data interception**: the observation of user data during a communication by an unauthorized user. This enables the intruder to slip between the communication partners, and modify the channel, unnoticed. The protection measures are the same as for identity interception.

- **Manipulation**: the replacement, insertion, deletion or misordering of user data during a communication by an unauthorized user. A checksum of the message, attached before encryption can prevent this kind of attack.

- **Repudiation**: the denial by a user of having participated in part or all of a communication. In this case, the signing of the message by the sender can prove the originator. A signature is the encryption of a compressed string of relevant data to be transferred.

Especially related to authentication, there are some additional threats to consider that are beyond the compromise of the communication channel, but rather related to the user preserving their prudence:

- **Compromise of the user's key**: one of the basic principles of authentication is that the user's secret key remains hidden to the outside–world. A number of practical methods are available for the user to hold his secret key in a manner that provides adequate security, like Smartcard, encrypted files, or floppy disks.

- **Forging of a certificate or ticket**: the malicious attempt of an attacker to construct some evidence to impersonate a user, thus misleading the communication partner about his identity. Consequently, it is very important that the authorities responsible for signing certificates are physical highly secure, trusted and well known to the public. Hence, a principle is able to verify the authenticity of a received certificate properly through the use of the public–known key of this authority.

- **Attack on the cryptographic system**: trying to break the system by the application of cryptoanalysis[1]. The likelihood of such an exposure should be

[1]The study and development of methods by which, without prior knowledge of the key, plaintext may be deduced from encrypted text, given sufficient working material and computational power. One aim of cryptoanalysis would be to discover the key, but that may not always be necessary[1].

negligible, and the designers of cryptosystems try to build in this property; at least from their current point of view.

2.2 Mechanism

To protect against perceived threats, various security services need to be provided. Besides access control, data confidentiality, and data integrity, peer entity authentication is the one of our concerns. The main goal of this service is to prevent impersonation, that is, the pretense to a false identity. It ensures that a user in a certain instance of communication is the one claimed. It can also be used to protect against replay of previously recorded messages. Authentication can either be single, where only one of the participants proves identity, or mutual, where both users authenticate each other.

The exchange of authentication information might be simple or strong. Simple authentication relies on the originator supplying its name and password, which are checked by the recipient. In contrast, strong authentication is based upon cryptographic techniques to protect the exchange of validating information. The data transferred is sent in encrypted form. Thus, this technique permits one to provide evidence that they know a particular secret without revealing anything about it.

The negotiation of which authentication mechanism is to be used takes place prior to the actual communication, and need not necessarily be performed via computer. A telephone call, or a face–to–face meeting might also be appropriate, in order to avoid compromising any information to the later used communication medium.

CHAPTER 3

MOBILITY

After looking at the purpose of authentication, there are several aspects to consider that are specific for the mobile environment and the mobile devices used within, i.e., laptops and palmtops.

3.1 Constraints

3.1.1 Hardware and Software

Mobile devices are far more vulnerable to physical attacks than their counterparts in a fixed network. Workstations and PCs are installed at one place, and are intended to stay there. Their disappearance is less probable and much more conspicuous than the theft of a mobile device, that can be stolen whenever it is left without surveillance. Besides the physical facts that impede the security of a mobile device, there exist also some hardware and software constraints:

- **Computing power**: due to their physical dimensions, mobile devices don't have the computing power comparable to PCs or even workstations. This implies that the overhead induced by communication and processing for additional security mechanisms, mainly authentication, should be kept as low as possible.

- **Limited battery life**: imposes careful usage of the power consuming units, such as CPU and memory. Therefore, overhead produced by the operations involved in the authentication mechanism, like encryption and storage of keys, should also be minimized.

- **Secondary memory**: the problem is, where to store the keys that the user needs for secure communication in a protected fashion, without exposing them

to an intruder who can gain unauthorized access to the mobile device. One possibility would be to keep the keys on a floppy disk, but then the user would have to carry the floppy disk around. A better approach to remedy this potential weakness, is to store the keys encrypted under a password, thus the user has an additional means of protection.

- **Type of operating system:** a multi–tasking operating system would be preferable. Thereby performing authentication in the background, without interfering the current running application.

3.1.2 Communication Medium

The limitations of the bandwidth suggest that the use of messages to and from the mobile unit be as small as possible, and the main work be shifted away from the mobile units; to the hosts in the fixed network. Another implication might be the preference of the datagram as a message vehicle. Therefore, it is not necessary to maintain a connection between two parties, which obviously induces some communication overhead. Moreover, datagrams eliminate the necessity of using session keys to provide a secure channel. Each single datagram can be individually protected against spoofing or corruption, thus excluding the possibility of an intruder taking over the connection after it has been established.

In addition, distortion by noise and interference, the limited bandwidth of the medium, as well as the unreliability of the link causes more delayed and lost messages than in a fixed network, which emphasizes the importance of the number of messages exchanged for authentication. This might be remedied in the future, when the currently predominant analog transmission medium is replaced by the next digital based generation with a higher reliability.

3.1.3 Security & Trust

Security is a major controversial issue in mobile computing, but presently, is largely ignored which is typical for a discipline in its infancy. Besides the high risk of a mobile unit to get stolen, the security of the operating system is its main weakness. DOS as the prevailing operating system for mobile devices, like laptops and palmtops right now, cannot be trusted due to a lack of protection mechanisms for user files, the user address space, and the possible recording of user action in the form of a log. This also implies the separation of the key generation from the remainder of the operating system; thereby establishing it as a trusted operation that is protected against penetration.

Mobile users are using resources, including software, at various locations. These resources may be made available by different service providers. Thus, a concept of trust[1] needs to be developed to allow mobile clients to use resources of different servers at different locations. In a mobile environment, control has to be exercised over the operations allowed to be performed by the different users at the different sites. The designated authorities are usually the authentication server. They must determine whether a user is granted access to the network services or not. This raises the question of how far a user can trust such a server in another domain. Also, how far a server is willing to trust an incoming user, especially with their different notion of trust. One domain might follow the policy to trust everybody they know personally, whereas other domains insist on a proof of their pretended identity.

3.2 IP for Mobile Computing

Right now, the researchers around the country, mainly the members of the IETF mobile–ip working group, are trying to establish a general protocol for the use of IP in the mobile environment. It compromises the two different approaches, based

[1]The notion of trust is based on the fact that an entity A trusts an entity B in some respect informally means that A believes that B will behave in a certain way

on the research done by Perkins and Bhagwat at the IBM Research Center [2], and John Ioannidis et al. at Columbia University [3].

This mobile host protocol should provide operational (i.e., users do not need to perform any special actions due to host migration) and performance transparency (the performance of a running application should be similar to that of the same application running on a fixed host). Operational transparency can be achieved by providing mechanisms to detect migration and to perform the appropriate actions to ensure continuing network services, whereas performance transparency needs optimal routing of packets to and from the mobile host (MH), as well as efficient use of network resources, such as transmission and processing bandwidth.

The Columbia protocol The fundamental concept in the Columbia mobile host protocol is its definition of a virtual mobile subnet spread across a number of real subnets. The mobile subnet exists wherever a Mobile Support Router (MSR) is located. The MSRs provide a gateway between the real subnets and the mobile ones. A MH is allocated a constant address on the mobile subnet. This means that higher layer protocols have an unchanging view of a MH's identity. The fact that the mobile subnet is spread across a number of real subnets gives the MH its mobility.

Figure 3.1 shows an example network, which illustrates a typical

environment with a number of Ethernet segments (IP subnets 2.x, 3.x) connected to backbone (1.x) via routers. Cooperating MSRs are attached to each segment defining two mobile subnets (4.x, 5.x) that are spread across the Ethernet segments. Also shown is MH1 and a fixed host (FH1). MH1 is physically connected to segment 3 and has already registered with MSR3 having identified it as its local MSR by listening for and responding to a *Beacon* packet that each MSR transmits at regular intervals.

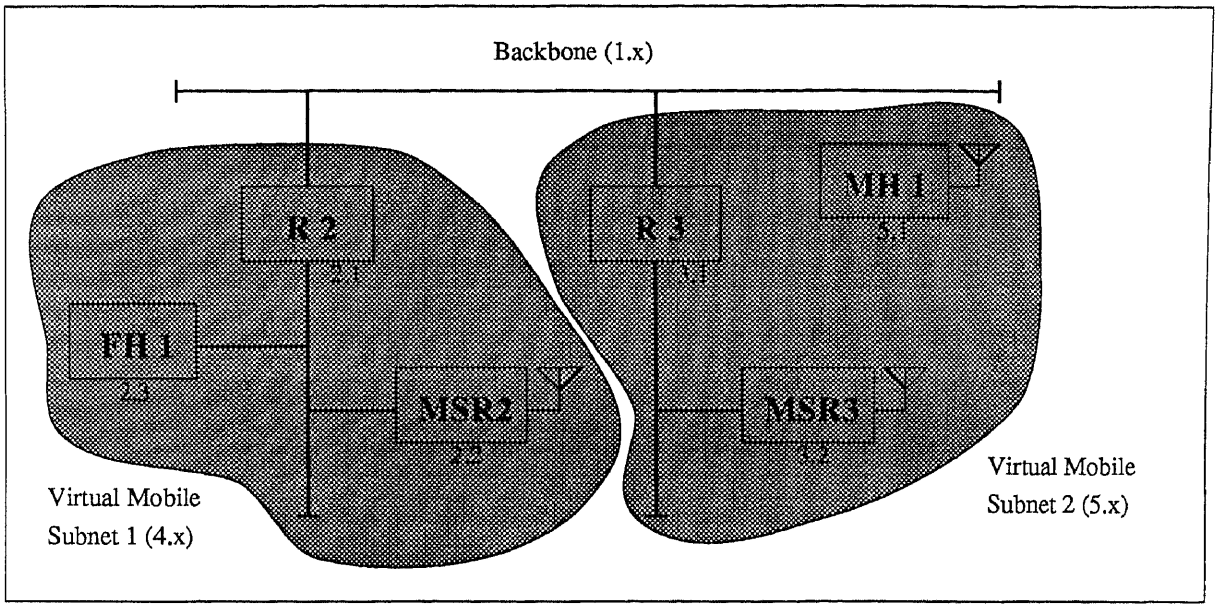

Figure 3.1 Example Columbia Network

If now FH1 wants to transmit a packet to MH1, the packet will initially be routed using existing routing protocols to the nearest router that advertises connectivity to the mobile subnet (probably MSR2 in this case). If MSR2 doesn't know the current location of MH1, it sends a broadcast packet to all MSRs asking who has MH1 registered with it. MSR3 will reply, and MSR2 tunnels any packets addressed to MH1, using an encapsulation protocol, to MSR3, which delivers it to the MH1.

If MH1 migrates now to segment 2, it registers with MSR2. MSR2 sends a forwarding pointer back to MSR3 informing it of MH1's new location. Future packets to and from FH1 will then be directly delivered, because the MH is now in the same subnet.

The IBM protocol The IBM protocol relies on TCP and UDP facilities to assist in the provision of mobility. The key idea in this concept is that each packet originating from a MH contains enough routing information that can be used by the remote host to send a reply back to the source along an optimal path. When TCP or

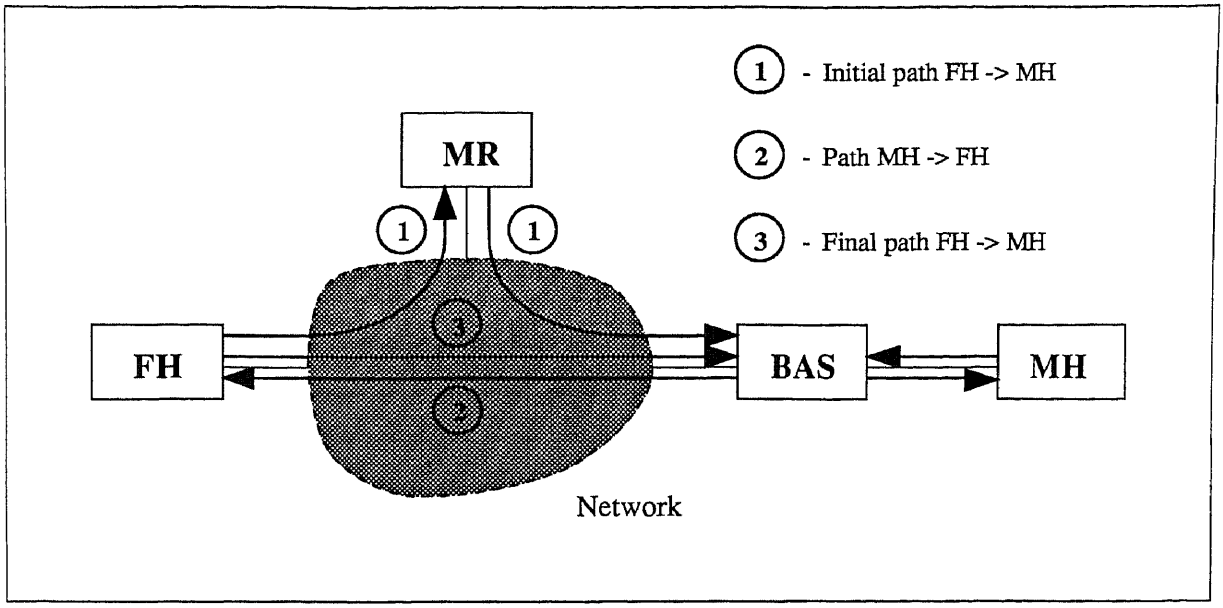

Figure 3.2 IBM protocol example

UDP receive a packet with a loose source routing (LSSR) option in the header, they send any packet in reply by the same path in reverse. These facilities allow existing hosts in the fixed network to participate in the routing of MHs. Whenever a MH connects to a network it must first register with a Base Station (BAS) that is also attached to the local subnet. A MH must also notify its Mobile Router (MR) of the address of the BAS (it is the MH's current location). A MH advertises connectivity to a group of MHs that it is configured to handle the existing routing protocols.

If a FH wants to transmit a packet to the MH, it transmits the packet normally and the packet is routed using existing routing protocols to the MH's MR. On receiving the packet the MR looks up the current location of the MH and inserts a LSSR option in the packet with the MH's current BAS as the first hop. The packet is then forwarded to the MH. Any return traffic from the MH to the FH specifies a LSSR option with the BAS as the first hop. When the FH receives this packet, it "learns" the MH's location, and sends subsequent packets directly to the MH's current BAS. This scenario is depicted in figure 3.2.

When a MH migrates to another subnet, it first registers with the new BAS, then notifies the MR of the new location and sends a message to the previous BAS telling it to delete its entry. Subsequent packets from the FH to the MH will be redirected by the original BAS back to the MR for correct routing until a packet from the MH to the FH forces the FH to route packets correctly via the new BAS.

3.3 Temporary residence

The basic assumption of user mobility is that a user has one *home*. A user's home is the administrative domain where he is registered on a long–term basis.

As a user migrates throughout an internetwork, he periodically pops up in a new, foreign domain. Regardless of the type of access, i.e., via wireless link or plugged into a fixed network, the goal of a mobile user is to obtain some service from the network. A user may simply be passing through a foreign domain or may be planning to linger about for some time. In either case, he must establish temporary residence in the foreign domain to obtain the service locally.

This process is similar to mobility in the real–world. For example, a person traveling from one country to another must engage in some form of a bureaucratic procedure with the purpose of establishing temporary physical residence at the new location.

This procedure can vary from country to country, as well as between different domains in a network. In most cases, it may be sufficient for a mobile user to possess a universal credential (passport/public–key certificate). Thus, a foreign domain can verify his identity. To get some information about the current status of a user, it might be necessary to interact with the user's home domain.

CHAPTER 4

KERBEROS

4.1 Introduction

Kerberos was designed for Project Athena at MIT[4, 5]. Kerberos, as well as SPX, represent client–server authentication mechanisms. Kerberos identifies clients of network services across an insecure network and protects the privacy and integrity of communication with those services. Its services are generally available to an application program through a programming interface. It is based on a symmetric cryptosystem[1], whereas SPX uses a public–key cryptosystem.

The computing environment consists of individual workstations privately owned and operated, similar to the laptops and palmtops in the mobile environment. Therefore, a workstation cannot be trusted to identify users correctly to network services.

The design is a refinement of ideas presented in Needham and Schroeder [6]. It incorporates trusted third–party authentication servers. The key idea for Kerberos authentication is the following: to use a service, a client must supply a ticket[2], previously obtained from Kerberos. The basic components include Kerberos authentication and ticket–granting servers (TGSs). A database contains information on each principal. It stores a copy of each principal's key that it shares with Kerberos. For a user principal, its shared *secret* key is computed from its password under use of some one–way function. Kerberos servers and TGSs read the databases in the course of authentication.

[1] In these cryptosystems, both sender and receiver must know the same secret key, which is used to both encipher and decipher a message. Nothing distinguishes the communication parties and, therefore, it provides a protected channel in both directions

[2] A ticket is a record that authenticates a client to a service; it contains the client's identity, a session key, and a timestamp, all of which is sealed by encryption using the service's private key.

Kerberos uses two main protocols. The credential–initialization protocol authenticates user logins and installs initial tickets at the login host. A client uses then the client–server authentication protocol to request services from a server.

4.2 Names

A principal, whether it is a client or a server, is the basic entity that participates in authentication. In most cases a principal represents a user or an instantiation of a network service on a particular host. As far as the authentication server is concerned, they are equivalent. A name consists of a primary name, an instance, and a realm, expressed as:

name.instance@realm

If the principal is a user, a genuine person, the primary name is the login identifier, and the instance is either null or represents particular attributes of the user, i.e., *root, admin.* For a service, the service name is used as the primary name and the machine name is used as the instance, i.e., *rlogin.myhost.* A Kerberos ticket is only good for a single named server. As such, a separate ticket is required to gain access to different instances of the same service. The realm is used to distinguish among different authentication domains; thus, there need not be one giant, and universally trusted, Kerberos database serving an entire company.

4.3 How it works

4.3.1 Getting the initial ticket

First of all, the user establishes a principal name and a private key, through some channel outside the system, for example, by walking up to the system administrator, and presenting his or her identification card. When the user now goes to a workstation, he logs into the system by using the credential–initialization protocol. The only piece of information that can prove his identity is his password. The initial

exchange with the authentication server is designed to minimize the chance that the password will be compromised. At the same time doesn't allow a user to properly authenticate himself without knowledge of that password. The protocol is specified as follows:

1. The user presents the workstation (client) with his principal name.

2. The workstation knows the name of its default realm. If not, another one is given by the user, and the complete name is composed. It then requests a ticket–granting ticket for the TGS from the authentication server (AS). This request is passed over the network in cleartext.

3. The AS looks up the name for the user and the TGS, finding the private keys for them in the database. It creates a temporary session key for use in the initial session between the client and the TGS, and prepares a ticket for the TGS.

4. The AS sends back the ticket and a copy of the session key encrypted under the key which is derived from the user's password.

5. The workstation asks the user for the password.

6. The workstation derives the key for decrypting the response from the AS from the typed–in password of the user. If it was correct, the workstation is able to decrypt the packet, and obtains the session key and the ticket for the TGS. It compares the timestamp in the response with the corresponding value in the initial request. If the response passes this test, the user knows for certain that the response was prepared by the Kerberos AS, because that is the only other entity in the universe that knows the user's private key.

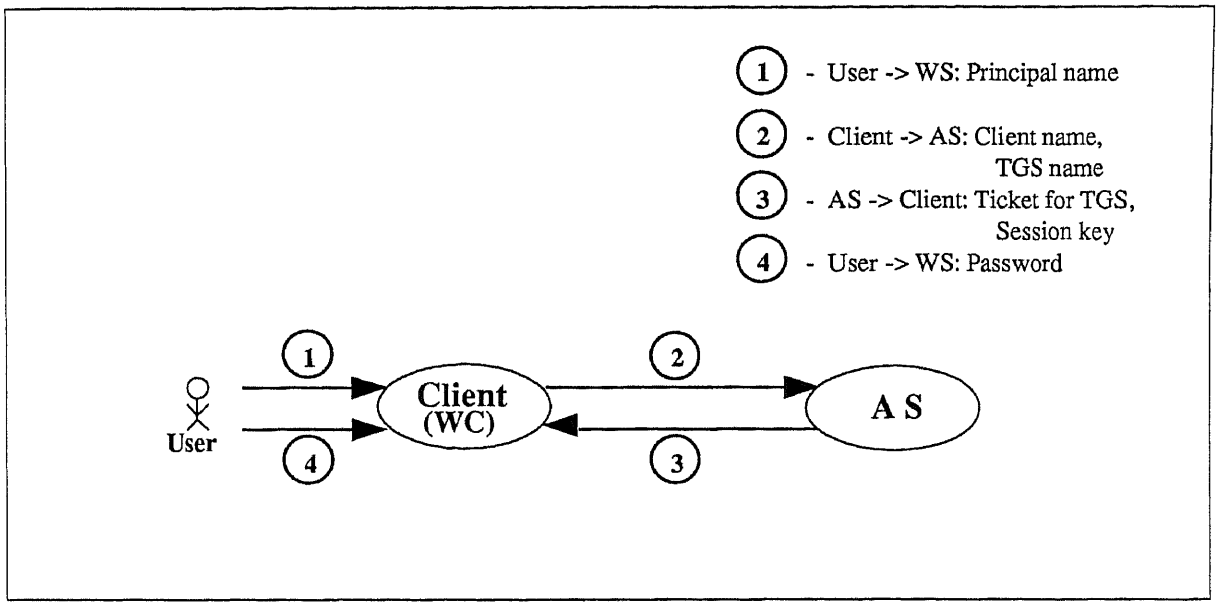

Figure 4.1 Getting the initial ticket

The exchange between the user, the client (workstation), and the AS is shown in Figure 4.1.

4.3.2 Requesting a service

After obtaining the initial ticket–granting ticket, a principal can request a server ticket from a TGS. The initial ticket is encrypted with a shared key between TGS and Kerberos.

Because a ticket is susceptible to interception or copying, it does not constitute sufficient proof of identity. Therefore, a principal presenting a ticket must also demonstrate knowledge of the session key included in the ticket. This proof is provided through an authenticator sent along with the ticket to the server. An authenticator is a simple mechanism designed to discourage attempts of unauthorized reuse ("replay") of tickets by someone who notices a ticket going by on the network and makes a copy.

The authenticator consists of, among other things, the client's name, network address, and the current time of day all sealed with the key that Kerberos issued for this session.

The basic steps of the client–server authentication are (see figure 4.2):

1. Client presents the ticket–granting ticket, an authenticator, and the the name of the desired server to the TGS.

2. The TGS goes through the same procedure as does any other Kerberos–mediated service. First it decrypts the ticket with its private key, and uses the temporary session key found inside to decrypt the authenticator. If all the authenticity checks verify correctly, the TGS knows the identity of the requesting client. It then looks up the service name in its database and finds the private key for that service.

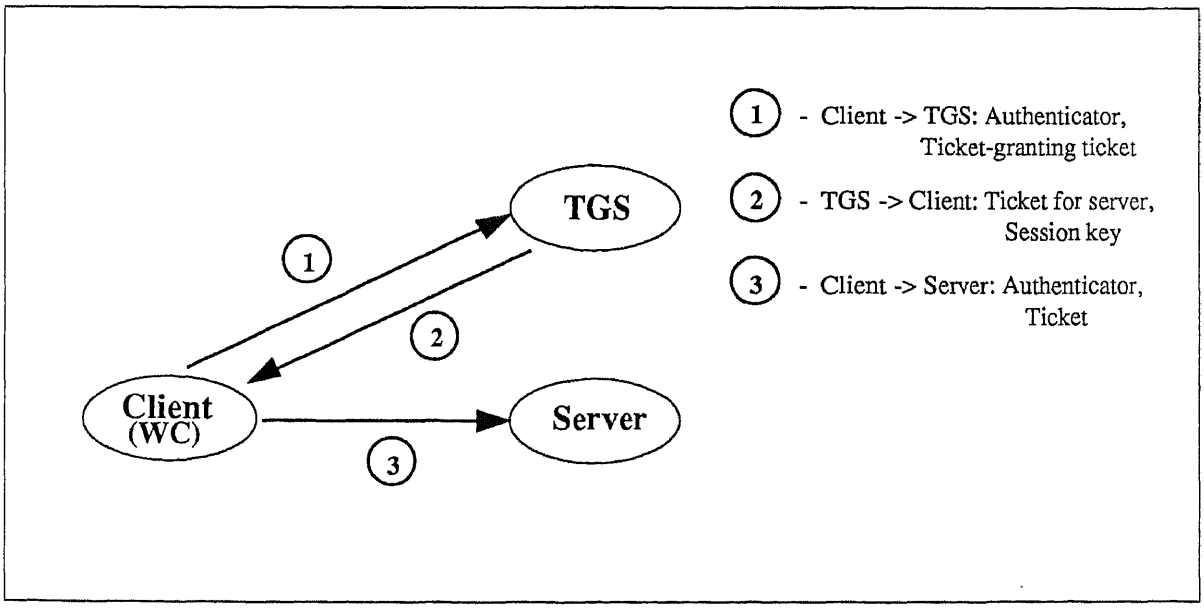

Figure 4.2 Getting a service ticket and requesting the service

3. The TGS creates a ticket with a new temporary session key for use between the client and the server. It is then sealed with the session key (obtained from the ticket in the request) and sent back to the client. This response is identical to the the form of the original response of the AS when it returned the ticket–granting ticket.

4. The client decrypts the response with the stored session key, and obtains the ticket for the desired service. The client is now able to present this ticket and a new authenticator to the server.

5. The server verifies the identity in a similar fashion as the TGS, by first decrypting the ticket with his private key. This ,simultaneously, proves the TGS as its originator. The server then checks the information in the authenticator on correctness. If the client's name and the network address of the incoming packet, in the ticket, and in the authenticator match, and the timstamps in the ticket and authenticator have not expired, then the request is taken as legitimate.

If the application requires mutual authentication, there would be an additional message sent in response to the client's request, in which the server adds one to the received timestamp, seals it with the session key obtained from the ticket, and sends it back to the client. This demonstrates that the server was able to read the timestamp from the authenticator, and hence that it knows the session key. It in turn is only available in the ticket, which is encrypted in the server's private key.

CHAPTER 5

SPX

5.1 Introduction

This authentication service is also intended for open–network environments [7]. In contrast to Kerberos, it is a public–key cryptography[1] [8] based authentication service. It shares many concepts and data structures in common with ISO/CCITT X.509 Directory Authentication [9]. SPX resembles in its functionalities Kerberos. It has credential–initialization and client–server authentication protocols. In addition, it has an enrollment protocol that registers new principals. It differs from Kerberos in the fact, that it eliminates the need for an on–line trusted authentication server. A certificate can be stored without any additional protection after it has been signed by a certification authority (CA). Moreover, it makes extensive use of hierarchical trust relationships, for naming as well as for the organization of the CA, which facilitates the scalability of SPX.

SPX has a Login Enrollment Agent Facility (LEAF) and a Certification Distribution Center (CDC) that correspond to Kerberos authentication servers and TGSs. LEAF, which is similar to the Kerberos authentication servers, is used in the credential–initialization protocol. CDC is an on–line depository consisting of public–key certificates (for principals and CA) and the encrypted private keys of principals. Note that the CDC need not be trustworthy because everything stored in it is encrypted and can be verified independently by principals.

SPX also contains hierarchically organized CAs, which operate off line and are selectively trusted by principals. Their function is to issue public–key certificates

[1]These cryptosystems involve a pair of keys, one of which is used for enciphering and the other for deciphering. One of the keys, known as the public key, is publicly known, whereas the other, the secret or private one, is kept private. Only possession of the complementary key enables deciphering of a messages enciphered under the other key. It is computationally infeasible to derive the secret key from knowledge of the public one.

(binding names and public keys of principals). Global trust is not needed in SPX. Each CA typically has jurisdiction over just one subset of all principals, while each principal trusts only a subset of all CAs, referred to as the trusted authorities of the principal. System scalability is greatly enhanced by the absence of global trust and on–line trusted components.

5.2 Names

The names in SPX for principals in certificates are X.500 [10] hierarchical names. X.500 names have a form unlike the simple string names used in most other naming schemes, for example, by the Internet Domain Name Service (used by Kerberos). An X.500 name is a sequence of "Relative Distinguished Names" (RDN's) that can also be considered as a path of pointers for navigating through a hierarchical database.

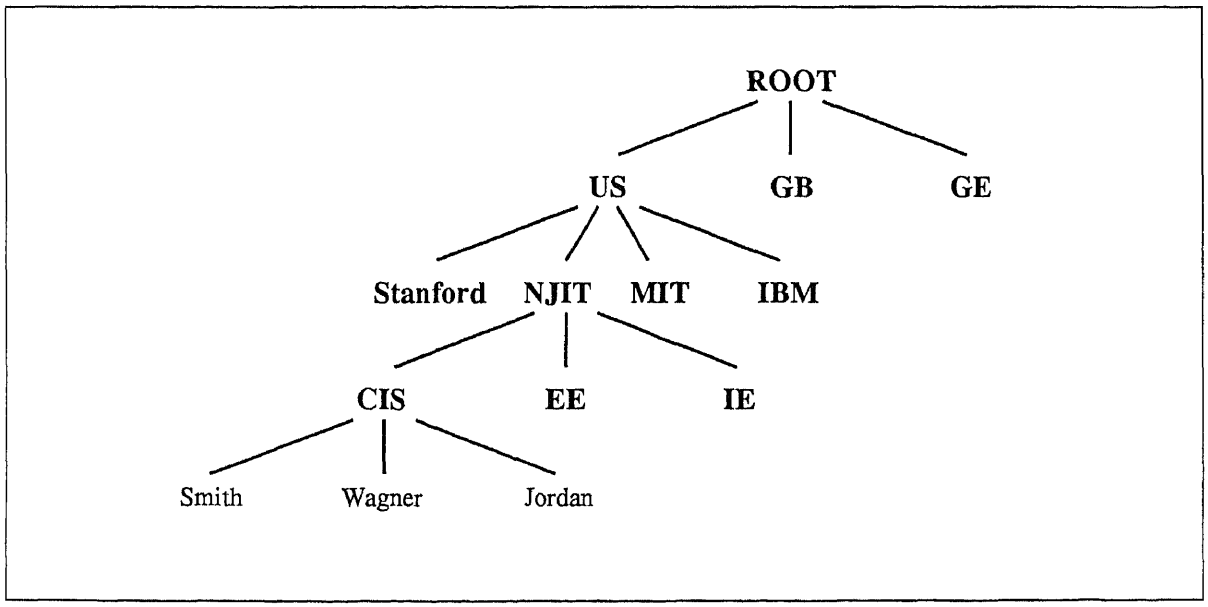

Figure 5.1 Sample of a naming tree (X.500)

The topmost level, after the root, accommodates the country of the principal described, followed by the organization. These two entries compose the domain prefix of the name. Next in the hierarchy, are the entries for the organizational unit, and

finally a common name for the principal. A sample naming tree is shown in figure 5.1. For example, a name based on this tree, that describes the user "Smith" in the organizational unit "CIS Department" of the organization "NJIT" in the country "US" would be denoted as:

/C=US /O="NJIT" /OU="CIS Department" /CN="Smith"

5.3 X.509

5.3.1 Certificates

A certificate contains the public key of a user, together with some other information. It includes an expiration date for the certificate, and the user's as well as the CA's (that issued the certificate) distinguished name, rendered unforgeable by encipherment under the secret key of a CA which issued it. It allows verification of the claim that a given public key does in fact belong to a given individual. Every user who knows the public key of a CA can retrieve the public key of another user by obtaining and decrypting the certificate corresponding to that user, issued by this CA.

A CA is an authority trusted by one or more users that creates and assigns certificates. It vouches for the binding of a public key to a user. Optionally, the CA may create the user's keys, since it is assumed to be physical secure, as well as running a trusted operating system. Thus, it is capable of generating keys in a protected fashion. On the other side, this involves transmission of the private key, which makes the key susceptible to interception.

5.3.2 The Hierarchy Structure

The CAs build up a hierarchical structure, similar to the one used by the naming scheme, further referred to as directory or tree, respectively. The higher level nodes

in this tree represent organizational or country–wide authorities. A CA can only guarantee the bindings of subordinate principals and CAs, and for its parent CA.

In order to authenticate another principal, a principal has to obtain the certification path from its location in the tree to the location of the principal seeking authentication. A certification path is defined by an ordered sequence of certificates of principals within the tree. Together with the public key of the initial principal, the sequence can be processed to obtain the public key of the final principal in the path.

It is possible to circumvent the hierarchical structure by using cross–certificates as either an optimization technique to shorten the path or to bypass an untrusted authority. These certificates must be established with bilateral agreement. An example would be a company communicating with a branch in Iraq, but not necessarily trusting the regime in Bagdad to reliable forward information. They would bypass the country–authority of Iraq, and would only include their own branch organization in the certification path.

5.4 Authentication and Key Distribution

Before a principal can participate in any authentication procedure, it has to install its certificate(s), after they are signed by a trusted authority, its encrypted private key, and a hash of its password in the CDC via the LEAF server. Thereby, they are generally available to SPX components when needed. In order for a principal to enroll itself in the CDC, it is necessary that the LEAF server have a trusted authority for the CA that issued the principal's certificate.

When a user walks up to his workstation, he initiates by entering his name and his password a login request to LEAF. LEAF reads the user's encrypted private key and the expected hashed password value from the CDC. If the one–way hash of the password from the CDC matches what was received in the request, then LEAF

returns the encrypted private key. The principal now generates a short-term key pair, and creates its login ticket structure. It contains a validity interval, the principal's uid, and the newly generated short–term public key, all signed using the long–term private key. Finally, the user gets his trusted authority certificates from the CDC. He uses the public key included in these certificates to verify the server certificates in subsequent authentication exchanges.

As in Kerberos, all authentication is done on behalf of principals. There exists a *claimant* principal, that is the entity seeking to be recognized as authentic, and a *verifier* principal trying to authenticate the claimant.

The approach to strong authentication taken in SPX makes use of the properties of public–key cryptosystems (PKCS). One of these properties is the interchangeability of the keys in the enciphering/deciphering process. This implies, that it is possible to encipher a message with the public key, and decipher it with the private key, and vice versa.

Authentication relies on each principal possessing a unique distinguished name (see section 4.2). Each principal is identified by possession of its secret key. A second principal is able to determine if a communication partner is in possession of the secret key, and can use this to corroborate that the communication partner is in fact the principal it pretends to be. The validity of this corroboration depends on the secret key remaining confidential to the user.

The authentication exchange is performed in the following way (see figure 5.2):

1. The claimant requests public key certificates for the verifier principal (1), and receives one or more certificates (2) depending upon the trusted authorities. In the simple case, the claimant is able to obtain a certificate that can directly be verified, because it has the public key of the authority that issued the certificate(s) for the verifier. In more complicated situations, the claimant may

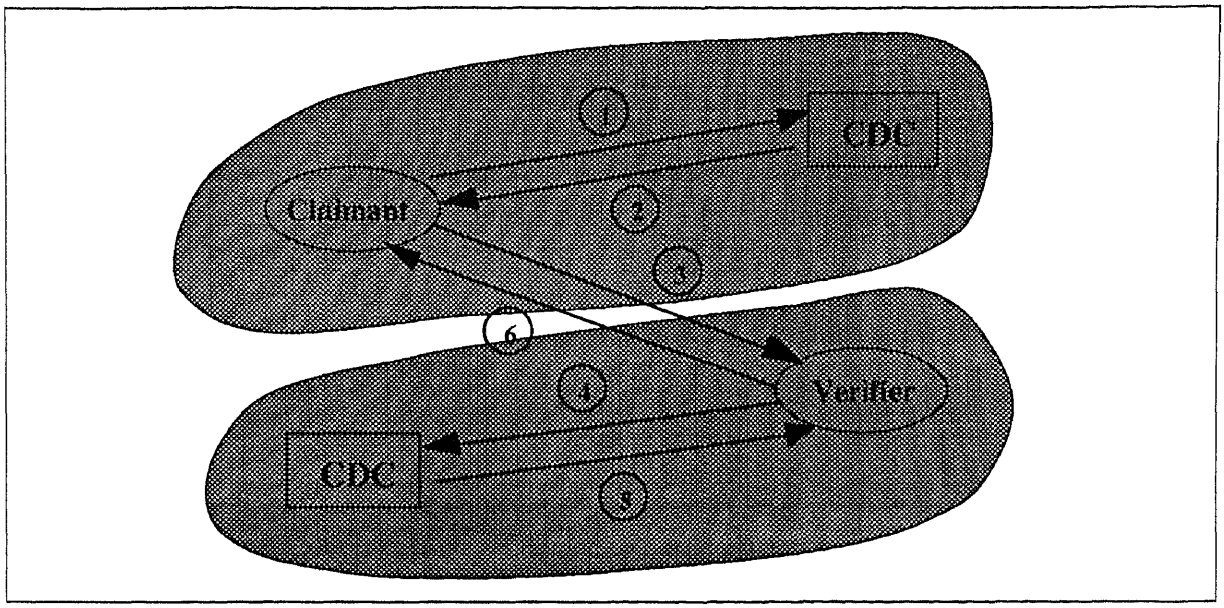

Figure 5.2 SPX authentication exchange

need up to two more certificates, namely a forward and a backward certification path to the verifier.

2. The claimant then generates a DES session key and encrypts it using the verifier's public key. It then generates an authenticator containing either a timestamp or a nonce[2], sealed with the DES key. If delegating is desired, the delegation key is also protected with the DES key, otherwise the claimant signs the encrypted DES key as a proof of its knowledge of the private delegation key without revealing it.

3. The claimant sends the authenticator, the delegator, and its ticket (created during the credential–initialization) to the verifier (3).

4. The verifier obtains the certificates it needs to verify that the ticket was created by the claimant from its CDC (4,5). It stores the delegation key, if required,

[2]A nonce is information that is guaranteed fresh, that is, it has not appeared or been used before. Perfect random numbers are good nonce candidates; however, their effectiveness is dependent upon the randomness that is practically achievable.

and unwraps the authenticator. In the mutual authentication case, the verifier returns the authenticator encrypted as a response (6).

Note that while an authenticator can be used only once, it is possible to re–establish the same authentication context multiple times. That is, the ticket and DES establishment components of the authentication token may have a relatively long lifetime. This permits a performance improvement, since repeated applications of public key operations can be alleviated. Therefore, one caches the authentication context, from a successfully used authentication token and the associated verified principal public key value.

CHAPTER 6

COMPARISON BETWEEN KERBEROS AND SPX

6.1 The Key System

The main, and most obvious difference between Kerberos and SPX is that they are based on two different cryptographic systems, SPX on public–key systems and Kerberos, in contrast, on secret–key systems.

The primary advantage of public–key cryptography is increased security: the secret key does not need to be transmitted or communicated to anyone; no one else needs to be trusted. In a secret–key system, there is always a chance that an enemy could discover the secret key while it is being transmitted.

Another major advantage of public–key systems is that they can provide a method for digital signatures. These signatures can be used as proof of the origin of the message, and thus authenticate the sender. Authentication via secret–key systems requires the sharing of some secret and sometimes requires trust of a third party as well, like the authentication server in Kerberos.

The main disadvantage of public–key cryptography is speed: there are popular secret–key encryption methods, as DES used in Kerberos, that are significantly faster than any currently available public–key method. SPX takes this into account by using a hybrid approach. The unique key distribution and authentication is designed based on a public–key system, whereas the subsequent communication is protected through session keys.

6.2 Naming

In version 4 of Kerberos, principals are named with three components: name, instance and domain, each of which may be up to 39 characters long. These sizes are too short for some applications and installation requirements. Especially for

big organizations which have structured their network into multiple subnetworks (domains) in a hierarchical manner. They maintain authentication servers for each of these domains, thus having to distinguish them from each other. They might not be able to form a meaningful domain name, since they can only identify each subnet with a short string in order to avoid exceeding the maximal string length.

This has been substituted in version 5 [11] by a principal identifier that is composed of multi-component names. The identifier is encoded in two parts, the domain and the remainder of the name, following the ASN.1 conventions. Therefore, it is possible to assign one string (component) to each subnet, and concatenate them to the domain name.

SPX uses distinguished names as identifiers for principals. The name is arranged in a hierarchical manner that imposes no restriction on length and characters used. It also facilitates the acquisition of a certification path due to inherent location information.

6.3 IP Dependence

The current running version of Kerberos, version 4, requires the use of TCP/IP as the underlying network protocol, since the decision whether or not a principal is considered as authenticated is basically based on the IP address of the workstation included in the authenticator as a proof of the origin of the service request. Moreover, the prevailing implementation of Kerberos, like FTP and TELNET, also assume TCP/IP as the underlying protocol. In version 5, this is partly remedied by introducing a type and length field in the message. Thereby, the recipient can interpret different network address types properly, however it still requires the use of a network address in the authenticator.

SPX is not as restricted on the use of any specific network protocol. Although it is based on TCP/IP right now, it doesn't use any location–dependent information

to prove identity. The possession of a private key corresponding to the public key retrieved from a CDC proves the identity of a principal, both the claimant's and the verifier's.

6.4 Certificates versus Session Keys

A ticket's extent is limited to the domain (realm) of its Kerberos server. This implies that a user entering a new domain has first to obtain a new ticket granting ticket for this domain, and then a new server ticket in order to receive a new session key for use with the services from the new domain. Therefore, there is a need of trust on both sides. Not only from the authentication server side in regard to the entrant, but also from the user in regard to the key distribution center in the new domain. The authentication server of the new domain has to verify the user's identity in order to believe him not to jeopardize the security of the network. However, the user has also to make sure that the session keys he receives are not compromised. Otherwise any subsequent communication is exposed to interception.

SPX doesn't need this kind of trust relationship. It overcomes the limitations of the trusted key distribution center of Kerberos by the use of RSA. As soon as a principal has obtained its private key from the CDC, there is no need to trust anybody else besides its trusted authorities. It can sign any message, send it to a communication peer, and the receiver can verify the origin of the message and the identity of the principal by simply obtaining the corresponding public key certificate from one of its trusted authorities.

6.5 Delegation

The "representation" of the user in the process is a set of credentials. These credentials include the user's identity as well as a secret cryptographic key that "speaks" for the user. That is, any process that can demonstrate that it possesses

this key may be taken as acting on behalf of the user. If a user process spawns another process on another machine, say, by doing a remote login, credentials may be needed on the remote system as well. They enable the remote process to do subsequent remote accesses on behalf of the user. The act of handing off credentials to another principal, so that the receiving principal can act on behalf of the initiating principal, is called delegation.

Delegation is not provided in Kerberos version 4. The remedy – manually requesting tickets on all remote machines – is inadequate as it requires the entry of a password, which would be transmitted unencrypted across the network. In version 5, it can be implemented through contacting the TGS with an additional ticket exchange and requesting a ticket valid for a different set of addresses than the ticket–granting ticket used in the request. However, this induces overhead due to the extra messages sent.

SPX can do this kind of representation transfer automatically, when needed (depending on the kind of security desired). The private delegation key that speaks for a user is transferred encrypted during the authentication phase.

6.6 Inter–domain Authentication

For inter–domain authentication, it is important to have global names and a trust relationship between domains. The latter one can be established by either sharing of a key between CAs that are willing to trust each other, or by installing the public keys of all trusted remote CAs in a local CA database, and introducing an inter–domain authority.

Kerberos can be used for authentication across administrative or organizational domains. It is based on the exchange of session keys between the involved domains' TGS, and the hierarchical based domain structure. A source domain is interoperable with a destination domain if it shares an inter–domain key directly with the

destination domain, or if it shares a key with an intermediate domain that is itself interoperable with the destination domain. Each domain exchanges a different pair of inter–domain keys with its parent and child node. When an application needs to contact a server in a foreign domain, it "walks" up and down the tree hierarchy toward the destination domain, contacting each domain's key distribution center. They provide a ticket–granting ticket to the next domain in the proper direction on the tree. When a ticket for the end service is finally issued, it contains an enumeration of all the domains consulted in the process of requesting the service. An application server can then decide, whether or not it wants to grant service based on the intermediate domain passed.

In contrast, inter–domain authentication in SPX is simply performed by the obtaining of the certification path to the appropriate server (verifier) by the client (claimant), and the presentation of the claimant's certificate combined with the corresponding return certification path, in order to verify the certificate, to this server. As far as obtaining tickets or keys is concerned, there is no interaction involved with any key distribution or authentication server in the intermediate domains. Since there is enough location information included in the domain part of the name of the desired service, the claimant can directly request the certificates for the service from the corresponding CDC in the remote domain. Once these certificates are retrieved and verified by the trusted authorities of the claimant, it can issue an authentication token to the remote server, who will be expected to call back the claimant's CDC to retrieve whatever certificates it needs, based on its own trusted authorities.

6.7 Summary

In addition to the points described in the previous sections, there exist some additional aspects (see APPENDIX A for a complete table). With respect to mobile

computing, and the constraints mentioned in chapter 3, SPX is the preferable choice as an authentication mechanism in this kind of environment.

The main reason for this decision has been the benefits gained by using certificates instead of session keys. It removes the need of trust relationships between domains. When a mobile user moves across domain boundaries, he does not need to obtain keys in order to establish a secure channel to perform the necessary authentication procedure. The encrypting of a ticket with the private key is enough of a proof of the sender's identity. Furthermore, this shifts also the main work away from the mobile host to the server (most likely in the fixed network), since the receiver of the encrypted message has to obtain the certificate for the sender. The receiver makes its decision, whether or not to grant a service to the requester, based on the validity of the ticket, decryptable with the public key included in the certificate. Therefore, it reduces the incurred overhead for the exchange of messages to and from the mobile host.

Another deciding factor is the absence of a delegation option in Kerberos. Although this is remedied in version 5, it induces additional message exchanges. Mobile users are certainly interested in delegating to a remote machines. Since the computing power and memory are limited on the mobile host, a user will use his mobile host as a "terminal" for a remote host. This could be either for performing work directly on that machine, or for connecting to other machines to transfer data or execute operations there.

The IP dependence of Kerberos is also an advantage to SPX. In contrast to a workstation, the network address of a mobile host does change while it is moving around. In the Kerberos protocol, the network address is used in the authentication procedure to prove the origin of the service request. Therefore, an assignment of a new or temporary network address in another network leads to the invalidity of the

ticket, and the denial of service. This, consequently, requires the request of a new ticket associated with every new network address.

Another disadvantage of the Kerberos protocol is its reliance on timestamps to prevent replay attacks. It includes the timestamp in the ticket and also in the authenticator. First of all, timestamps require roughly synchronized clocks, in order to work properly. Therefore, there would have to be a coordination of the clock synchronization in different domains, which is unlikely. In addition, the unreliability of the wireless link, if used, may cause an increased delay or loss of messages, leading to tickets and authenticators to be considered stale and not acceptable any more. The use of a nonce, as in SPX, removes this dependence on synchronized clocks, while still preventing replay of previously recorded messages.

Furthermore, the lifetime of the tickets in Kerberos is limited. This raises the problem of how long should a ticket be valid? A short lifetime is inconvenient, because every new request for a ticket–granting ticket requires the entry of the password. On the other hand, long–lived tickets are vulnerable to capture. This holds especially in the mobile environment, where any intercepted ticket can be used to impersonate the user after he leaves the cell. In SPX the principal creates the ticket for itself, and thus is able to adjust the validity to the changing requirements of the different environments.

Last, but not least, it is planned to incorporate the certificate distribution of SPX in the future in an ubiquitous directory service (i.e., X.500). Hence, it will then be possible to retrieve certificates in the same manner as already implemented for the other attributes of a principal, like the name, organization of a user, his telephone number, etc. .

CHAPTER 7

ADAPTATION OF A MECHANISM

7.1 General

In order to establish an authentication mechanism in the mobile environment, it is necessary to understand how such a mobile network is structured, how messages are exchanged, and how a user moves around. Furthermore, it must be considered how far the needed facilities for such a mechanism can be incorporated with existing features and properties.

Among the possible network topologies, the simplest and probably the most common architecture, is that of a static infrastructure forming a tree, with "cells" where the mobile units (MUs) are attached at the leaves. The intermediate nodes represent some kind of authority or database that keep track of the MUs in the cells located below them. They cover, for example, all cells of a company's area. Each cell is a logical or physical area where MU can talk to the base station(s). That is, the machine(s) that route traffic from the MUs in the cell to the rest of the network and vice versa. The base stations are usually the endpoints of the fixed network. MUs in the same "cell" may or may not be able to communicate directly with each other; they may have to use the cell controller as a hub. It should be possible to incorporate in this topology the hierarchical structure of the certification authorities in order to combine the necessary registration in a new entered cell with the purpose of authenticating the MU.

Naming and addressing are two very important, interrelated problems. The problem is already complicated in a static world, but gets even more so by adding the ability to move one or both endpoints of a connection while the connection is active. In a static network, the address of a host is administratively determined by its location. In a mobile network, the addresses of the network's entities have to be

maintained and updated as the entities change locations. There must be a database, be it centralized or distributed, about the location of each MU, and algorithms for computing routes to them.

The system also has to know where each MU is. Eventually, the MU will have to reveal its location, but this can happen in two possible ways: the MU can notify the system every time it moves (*active sign–on*) and the system can then, immediately or lazily, update its databases; another possiblility is that the system can seek out the MU when it needs to talk to it (*paging*).

Before the network can route data to mobile stations (and, in most cases, before it can receive data from them), the MU must register with the network. This registration may serve either or both of the following purposes:

- Establish a location for the MU (for routing purposes).
- Identify and authenticate the MU (for security and accounting purposes).

The location establishment is beyond the scope of this thesis, and the authentication procedure depends on the assumed scenario. Registration may occur when the MU first comes on–line, finds a "base station", and handshakes with it. The base station then has the option of storing the location information locally, propagating it to a centralized database, or to other, possibly neighboring base stations. Propagating information facilitates the verification of the user's identity and the assessment of his trustworthiness. The receiving base stations can also provide additional information about experiences and incidents with the user. Keeping this location information has the advantage that the network knows where the MU is, when it needs to contact the MU.

As far as the user is concerned, the incorporation of the authentication mechanism should fulfill certain criterias:

1. Be as transparent as possible
2. Have as little as possible influence on the performance
3. Preserve user confidentiality

Except the initial log–in, the user should not need to perform any additional action to authenticate himself to a service or a foreign domain. This is mainly achieved by the credentials, a user carries with him. When requesting a service or entering a new domain, the information included in the credentials, in particular the private key of the user, is used as proof of the user's identity.

Regarding performance, there are two aspects to consider. First, the delay incurred by the process of handing–off a user from one domain to another, and second the computing power and memory used by the authentication procedure. This includes operations like encryption, key generation, and keeping the state during the authentication exchange. There is a tradeoff between these two aspects. Keeping less state information imposes larger messages, and thus a greater delay. On the other hand, keeping more state information requires more memory, but reduces communication for the smaller messages. Computing power and memory can be saved by shifting as much as possible of the workload to the fixed network. In addition, it is preferable to use for example signatures which need significantly less computation than the full encryption of a message.

In order to provide user confidentiality, user identification information must be protected from disclosure. This is especially important for the user's credentials, since everybody in possession of the private key can impersonate the user. It can be

done with the use of a device like a Smartcard[1], and by exposing as less as possible of this vulnerable data to the communication medium.

7.2 Possible Scenarios

7.2.1 Movement within one cell

The easiest scenario to handle is the one when a mobile user is only moving within one cell without leaving it. The authentication procedure is similar to logging into a workstation, except that it might have to cope with the problems of the wireless link, if used. Distortion and noise can cause loss or delay of messages expected by a communication peer. Hence, the timestamps in the ticket can become stale, leading to the denial of the requested service. Besides the prevention of replay attacks, the coping with stale tickets is the main reason for using a challenge–response exchange[2] with nonces instead of timestamps. In order to determine whether or not a message including a nonce has to be considered stale, the party who created the nonce keeps a record about it that also includes a validity interval.

Analogous to the workstation environment, the mobile user is authenticated, not the machine he is sitting at, since the machine itself doesn't provide a high means of protection against penetration. There is usually only the password that prevents access to the machine. Moreover, a mobile unit, be it a laptop or a palmtop, is much more vulnerable against theft, merely due to its physical dimensions.

It must be distinguished whether the user logs into his home network or into a foreign network. Presently, without the possibility to retrieve the user's certificate

[1]A Smartcard is a small device incorporating a processor and memory which can be carried by the user for identification. It can hold authentication–related data and the user's credentials protected by a strong key. The user can activate it by entering of a PIN.

[2]In a challenge–response exchange, one party generates a random number, and sends it over to its communication peer. The peer encrypts the random number with its private key, which is a sufficient proof of its identity, then it is sent back. The originator of the random number can verify the identity of its peer by decrypting the response with the corresponding public key.

from an ubiquitous directory service, the base station of a foreign network has to interact with the home domain of the user in order to provide him with his initial credentials. This might incur a considerable overhead. Thereby, slowing down running applications, since they have to wait for the authentication procedure to complete before they can proceed in the foreign domain. A way to remedy this problem would be to forward the credentials from one base station to the next one. They could be conveyed within the location–updating exchange from the preceding base station to the current one. A better solution, also with respect to user confidentiality, would be the use of a Smartcard or a similar device. Hence, the user can carry all his security–related information with him in a secure fashion, making use of them on demand.

Besides that, when a user has received his credentials, and wants to authenticate himself to a service, he does not need to initiate any specific actions different from the ones he would perform in the a fixed network (see section 4.4. for details).

After the user is authenticated in one cell, there should be no additional authentication involved while he is moving around. Due to the higher vulnerability of a mobile unit, it might be feasible to reinsure every now and then that the user is still the same as authenticated. This could be possible by simply encrypting the beacon, frequently sent by the base station, or responses to the user, with the user's public key. Therefore, only the user in possession of the private key is able to decrypt it.

7.2.2 Movement between cells, but within the same domain

The next step toward a complete registration of mobile users wherever they are located, as far as authentication is concerned, is the consideration of the movement between cells, but still within the same domain.

A domain has a physical as well as a logical dimension. The physical dimension is the area covered by the cells it is composed of, whereas the logical one depends

upon factors like an organization boundary. As long as a user stays within the same domain, he can assume that the security policy and the authentication mechanism remains the same. For example, a company will trust an employee to be the one he claims to be, no matter where he is located on company grounds.

This implies that an extensive authentication with establishment of a temporary residence is not necessarily needed. When a user enters a new cell, he has to register with the base station of this cell in order to obtain services from it. He has to identify himself for accounting and security purposes. The base station of the new cell determines the degree of authentication needed in order to grant services to the user. In certain circumstances, like when entering an area with increased security requirements, it might be necessary for the base station to interact with the user's home base in order to authenticate him properly. However, usually the certificate of a user should be enough proof of his identity. This follows the idea to avoid authentication, if possible. It saves the overhead incurred in a complete authentication procedure, both communication costs, computing power and memory consumed.

To prevent against replay of tickets from a user who has already left the cell, it is favorable to use a challenge–response exchange instead of including a timestamp, and recording previously received tickets. This is especially important during the time a user is in transit from one cell to the other, until he pops–up in a new one.

7.2.3 Movement between different domains

Before any authentication is possible in a foreign domain, there has to be an agreement about the authentication mechanism used. This agreement can either be reached by prior negotiation, or by following a policy set by some higher level authority.

There is a difference between a user popping–up in a foreign domain and one moving between two adjacent foreign domains. While a user makes his way from

one domain to another, his credibility must be confirmed with every crossing of domain boundaries. There has to be a hand–off from one domain to the next one, including the user's state, current session activities and authentication information. To facilitate the obtaining of information about the user, the base station can refer to the authority in the preceding domain instead of interacting with the user's home domain every time.

In general, we can not assume that the path taken by a mobile user is continuous. That is, a user may operate in one domain and several hours later pop–up in another domain thousands of miles away, e.g., an eight hour flight from Germany to the United States carries a user between two distant, non–adjacent domains. This case is similar to the one when a user initially pops–up in a foreign domain, with the difference that he has already obtained his credentials.

When a user first contacts the base station of a cell in the foreign domain, the base station will recognize him as coming from another domain. The new domain needs to establish temporary residence and some form of address convention for the entrant in order to send him messages. In contrast to the previous scenario, a base station in a new domain is certainly interested to learn about the user's identity. Instead of the original authentication exchange that is initiated by the user, this time the base station, as the contact point to the network and thus service provider, initiates the exchange. It sends some kind of unique value to the user. The user adds his name, encrypts it with his private key, and sends it back to the base station. The base station obtains the certificate for the user, and verifies his identity by decrypting the response from the user. When the base station decides to grant service to the mobile user, it also provides him with its certificate issued from one of the user's trusted authorities. Thus, the user can verify the identity of the network entered.

If the base station is not satisfied with the information provided by the user, it has to contact the home authority of the user. In order to obtain information

about the user from his home domain, it might be necessary to traverse intermediate networks. When there are some untrusted networks, it is possible to circumvent them by the use of cross–certification paths. They can also be used to optimize frequently occurring authentication between domains. Other means of information retrieval about the user are described in section 7. 2. 1 (forwarding and Smartcard).

CHAPTER 8

FUTURE WORK AND CONCLUSION

This thesis presented a study about the incorporation of authentication in mobile computing. It included a comparison of the two predominant authentication mechanisms, namely Kerberos and SPX, with respect to the mobile environment. It has taken into account the limited computing power and memory of the mobile devices, as well as the problems induced by the unreliability of the communication medium, at least when wireless communication is involved.

After considering the advantages and disadvantages of both mechanisms in regard to different areas, SPX seems to be the preferable choice in this kind of environment. This selection is mainly based on the increased security, the removal of the trust relationships between domain by the use of certificates, non–IP dependence, the use of nonces instead of timestamps, and the possibility of delegation provided by SPX.

Due to the fact that the adaptation of SPX in the mobile environment is only based on theoretical background, it has rather to be seen as suggestions of what can be done, and what might be the changes necessary in the protocol. It doesn't claim to be any kind of implementation.

Further work must certainly include an implementation in order to verify the assumptions made about the environment and the devices. It has to be shown that the mobile devices and the communication medium are already capable of performing the authentication procedure. The incorporation of the authentication procedure shouldn't penalize the performance of other application. Furthermore, the procedure of authenticating a user while moving around should, with the exception of the initial login, be transparent to the user.

APPENDIX A

COMPARISON OF KERBEROS AND SPX

Area	Kerberos	SPX	Preference
Cryptosystem	Sekret–key	Public–key	None
Naming	3–component Name	Distinguished Names	SPX
IP dependence	IP network address in authenticator	Independent on under–laying network protocol	SPX
Subsequent authentication	New session keys	Certificates	SPX
Delegation	Additional message exchange	Conveyed in authentication exchange	SPX
Inter–domain authentication	Exchange session keys between domains	Resolve certification path	SPX
Ticket–Lifetime	Restricted to 21 hours	No restriction	SPX
Timestamps	In Authenticator	Substitutable by nonce	SPX
Encryption mechanism	Only DES	RSA or any other	SPX

REFERENCES

1. D. W. Davies and W. L. Price. *Security for Computer Networks.* Wiley Series in Communication and Distributed Systems. John Wiley & Sons, second edition, 1989.

2. Charles E. Perkins and Pravin Bhagwat. "A Mobile Networking System based on Internet Protocol(IP)". In *Proc. of the USENIX Mobile & Location-Independent Computing Symp.*, pages 69–82, Cambridge, Massachusetts, August 1993. USENIX.

3. John Ioannidis, Dan Duchamps, and Gerald Q. Maguire. "IP–based protocols for mobile internetworking". In *Proc. of SIGCOMM '91*, pages 235–245. ACM, September 1991.

4. J. G. Steiner, C. Neuman, and J. I. Schiller. "Kerberos: An Authentication Service for Open Network Systems". In *Proc. Winter Usenix Conf.*, pages 191–202, Berkeley, Calif., 1988. Usenix Assoc.

5. S. P. Miller, B. C. Neuman, J. H. Saltzer, and J. I. Schiller. "Section e.2.1: Kerberos Authentication and Authorization System". Technical report, M.I.T. Project Athena, Cambridge, Massachusetts, December 1987.

6. R. M. Needham and M. D. Schroeder. "Using Encryption for Authentication in Large Networks of Computers". *Communication of the ACM*, 21(12), December 1978. pp. 993-999.

7. Joseph J. Tardo and K. Alagappan. "SPX: Global Authentication using Public Key Certificates". In *Proc. IEEE Symp. Research in Security and Privacy*, pages 232–244, Los Alamitos, Calif., Order No. 2168, 1991. IEEE CS Press.

8. R. Rivest, A. Shamir, and Adleman. "A Method for obtaining Digital Signatures and Public–Key Cryptosystems". *Communication of the ACM*, 21(2), February 1978. pp. 232–244.

9. ISO/CCITT. "Information Processing Systems – Open System Interconnection – The Directory Authentication Framework". ISO/IEC 9594–8, 1989. also CCITT 1988 Recommendation X.509.

10. ISO/CCITT. "Information Processing Systems – Open System Interconnection – The Directory – Information Model". ISO/IEC 9594–1, 1989. also CCITT 1988 (blue book) Recommendation X.501.

11. J. T. Kohl and B. C. Neuman. "The Kerberos Network Authentication Service". Version 5 revision 5, M.I.T. Project Athena, Cambridge, Massachusetts, April 1992.